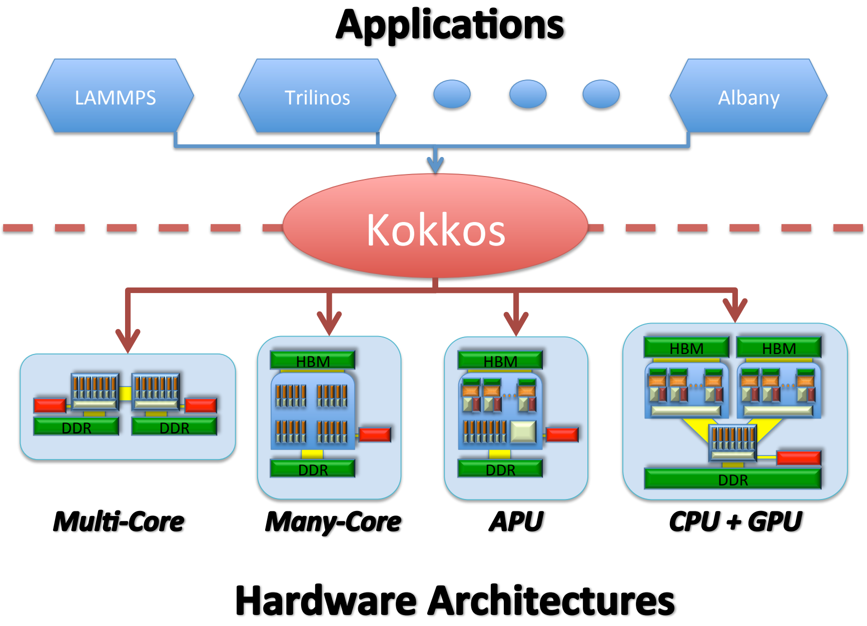
Kokkos: Enabling Manycore Performance Portability for C++ Applications and Libraries

Christian R. Trott and H. Carter Edwards
Scalable Algorithms, Sandia National Laboratories

Supercomputer architecture trends point to a decreasing number of distributed-memory compute nodes where each node has an increasing number of shared-memory cores. Furthermore, modern high performance computing (HPC) nodes have diverse and heterogeneous types of cores and memory. For applications and domain-specific libraries/languages to scale, port, and perform well on these next generation architectures, their on-node algorithms must be re-engineered for thread scalability and performance portability. The Kokkos library helps HPC applications and domain libraries implement intra-node thread-scalable algorithms that are performance portable across diverse manycore architectures such as CPUs, Intel Xeon Phi, and GPUs through its core abstractions. Through its abstraction layers it isolates application developers from the ever-changing details of the hardware architecture of coming high performance computing platforms.



Kokkos' six key abstractions were developed to transparently manage thread parallel computations' data access patterns. This capability enables users' thread parallel computations to be performance portable across diverse and heterogeneous manycore architectures such as multicore CPU, Intel Xeon Phi, and NVIDIA GPU. These abstractions are designed to accommodate other architectures such as AMD

APU and the anticipated evolution of Intel Xeon Phi and NVIDIA GPU. Kokkos' six key abstractions are as follows.

(1) Users' express parallel computations with **parallel patterns**; e.g., for-each, reduce, scan, and directed acyclic graph (DAG) of tasks.
(2) Parallel computations occur within **execution spaces** of a heterogeneous architecture; e.g., latency-optimized CPU cores and throughput optimized GPU cores.
(3) Parallel computations are scheduled according to **execution policies**; e.g., statically scheduled range [0..N) and dynamically scheduled thread teams.
(4) Data are allocated within **memory spaces** of a heterogeneous architecture; e.g., in CPU main memory and GPU memory.
(5) Data are allocated through multidimensional arrays with **polymorphic layout** that specifies how an array?s multi-index domain space is mapped to an allocation within a memory space.
(6) Arrays may be annotated with **access intent traits** such as ”random access” or ”atomic access.” Kokkos may use these traits to map array entry access to architecture-specific mechanisms such as GPU texture cache or atomic instructions.

Thesis: Using the 6 Kokkos abstractions it is possible to write applications which are *performance portable* across diverse hardware architectures.

The Kokkos library API has been designed, implemented, tested, evaluated, and improved through many development iterations. Given an understanding of Kokkos' abstractions and an intermediate knowledge of C++ (2011 standard), this library API is concise and intuitive for users. Performance portability was demonstrated across architectures using miniApps including MiniFE, MiniMD, MiniAero, and Lulesh.

Kokkos is at the core of many projects at the Sandia National Laboratories to move applications to next generation architectures. It is now the default backend for the second generation solver stack in Trilinos, it is used to move LAMMPS forward, and Kokkos is central for the majority of ATDM efforts. Furthermore initial experimentation with Kokkos in Sandia's production codes is ramping up. With that the range of applications using Kokkos includes simple linear algebra, solvers, molecular dynamics, graph analytics, finite element codes and fluid dynamics.

Development of Kokkos is now taken place on github (github.com/kokkos/kokkos) with two full time developers and a number of part time supporters. The project is open source under the BSD license.

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy?s National Nuclear Security Administration under contract DE-AC04-94AL85000.